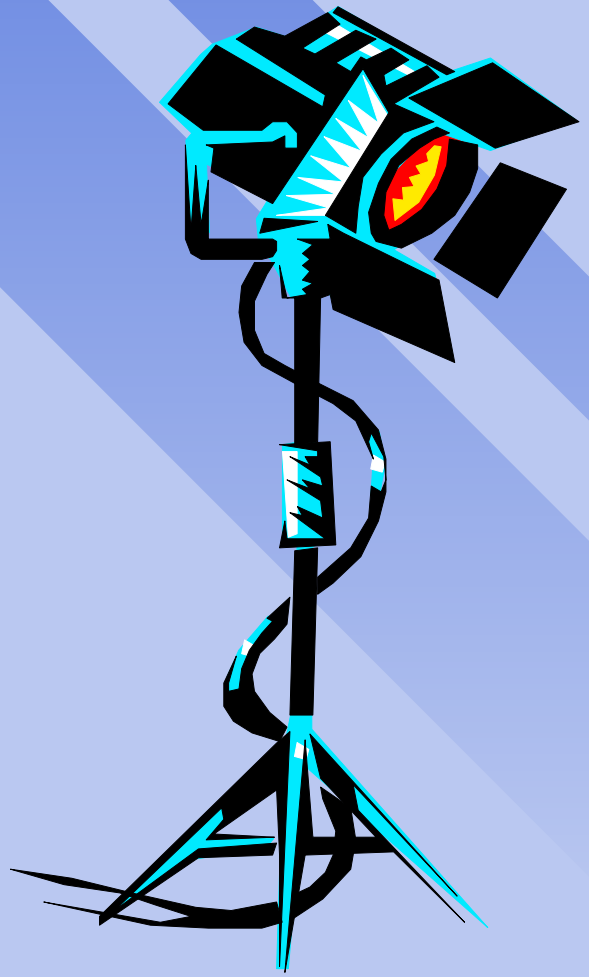
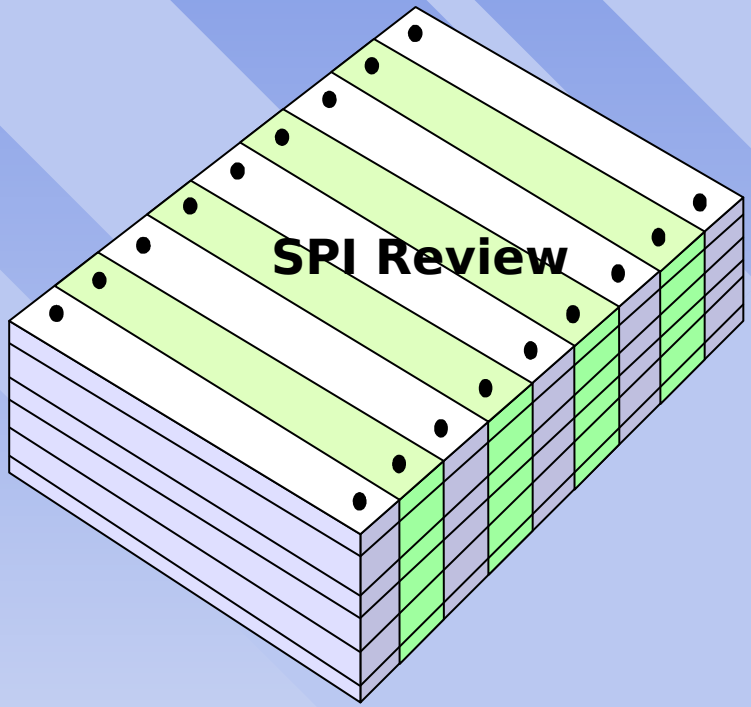
# Welcome to the
# System Modification Scenario
# Detailed Unit Test
# Training Class

# Course Objectives:

- High level review of SPI, CMM and SMS
- Introduce Unit Testing (UT)
- Identify UT tasks, subtasks, and procedures as outlined in the SMS

# SECTION 1





# SPI, CMM, SMS Review

- The student will answer questions about:
  - Software Process Improvement (SPI)
  - Capability Maturity Model (CMM)
  - System Modification Scenario (SMS)
- The student will identify:
  - The Key Process Areas of CMM Level 2
  - The outline levels of the SMS




# Software Process Improvement (SPI)

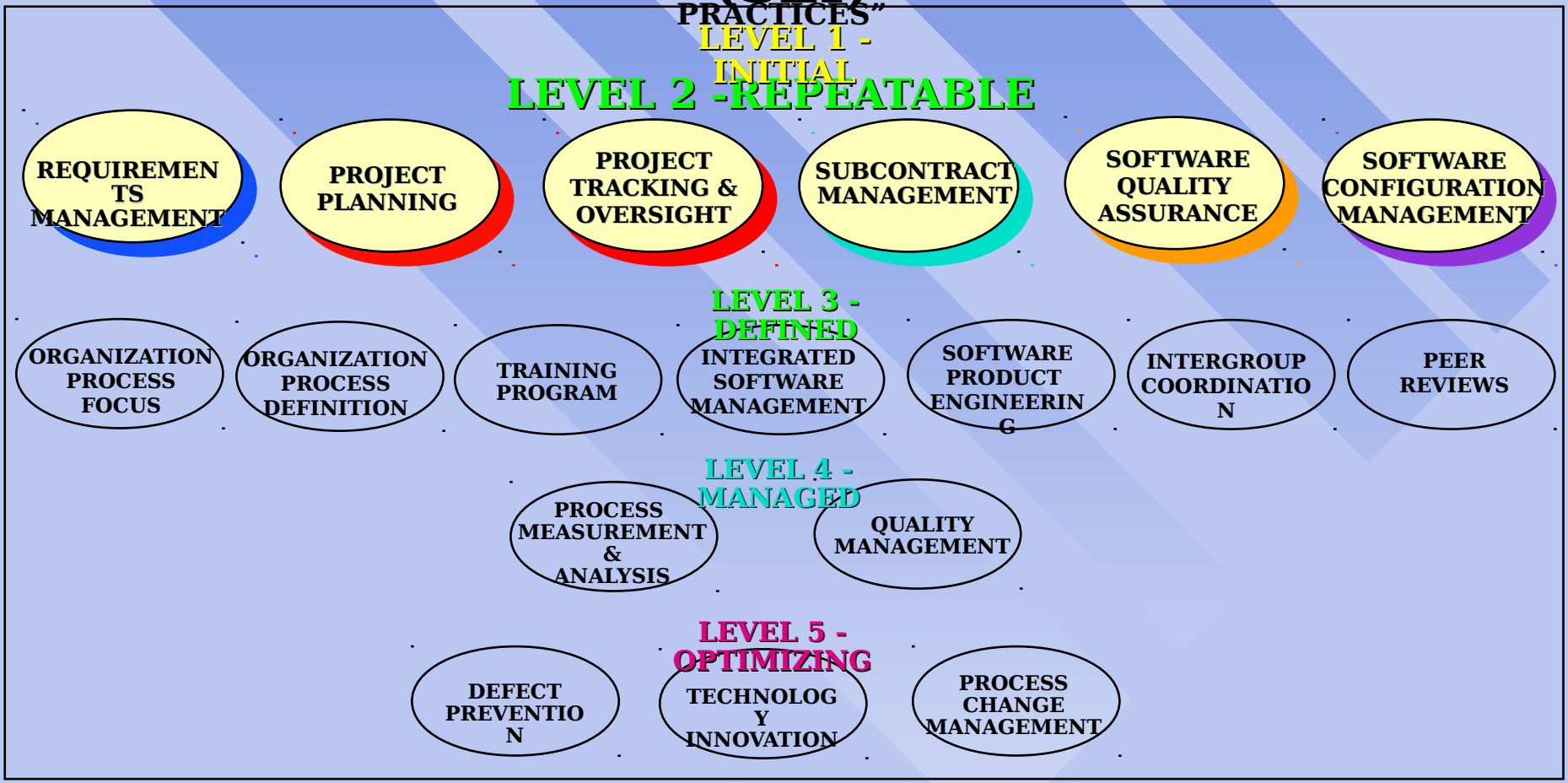
- Effort to Improve Software Process
- System of:
  - Tasks
  - Tools
  - Standards, Methods, Practices
- Applicable Throughout Software Life Cycle




# Capability Maturity Model (CMM)

- A FRAMEWORK FOR EFFECTIVE SOFTWARE PROCESSES
- Identifies:
  - Maturity Levels
  - Key Process Areas
  - Common Features

# KEY PROCESS AREAS (KPAs)
## As Defined by
## SOFTWARE ENGINEERING INSTITUTE (SEI)
### "SOFTWARE ENGINEERING AND MANAGEMENT PRACTICES"



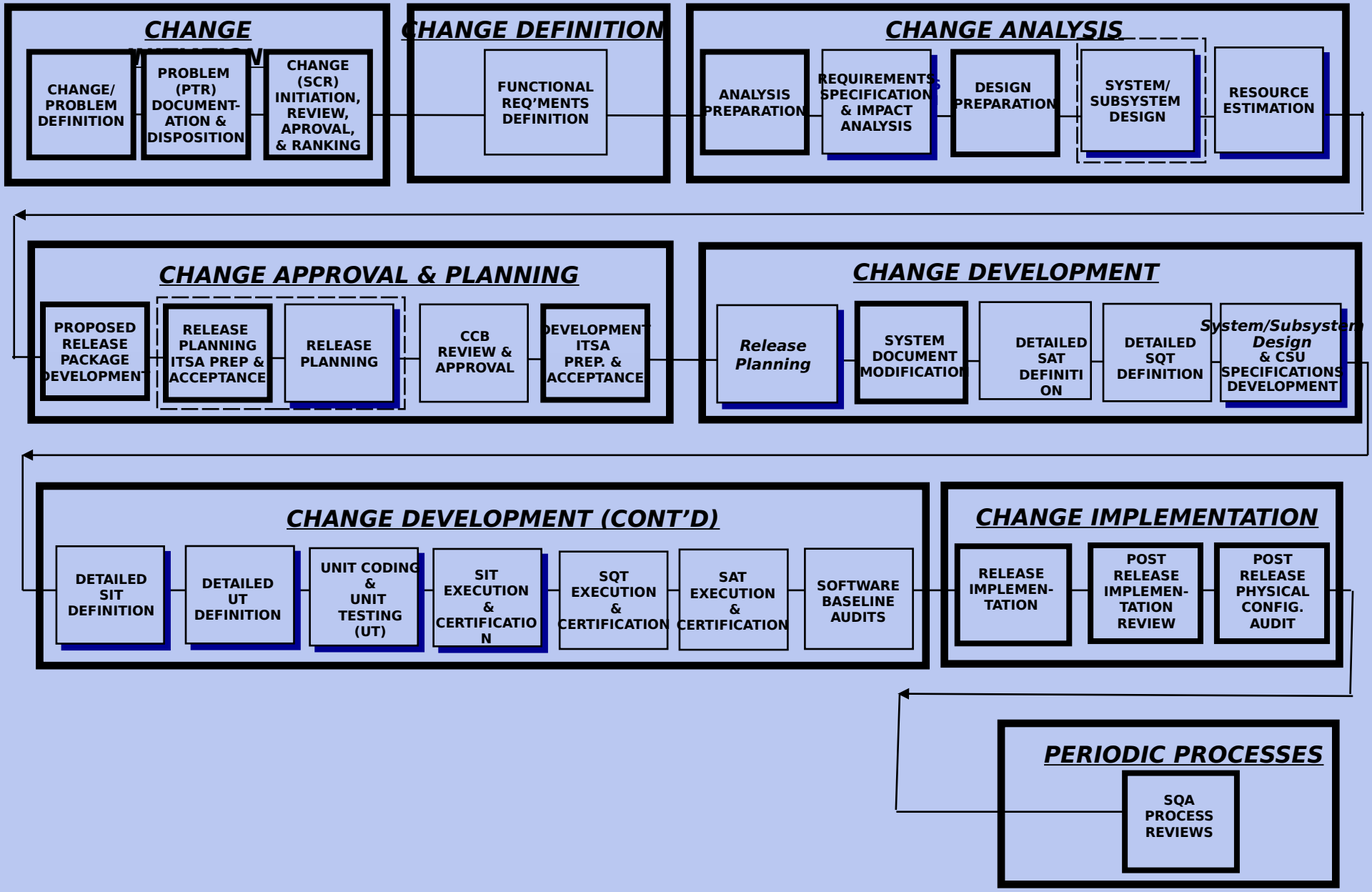



# System Modification Scenario (SMS)

- One part of the Software Process Architecture
- Focuses on routine system modification
- Provides:
  - Process definition
  - Description
  - Documentation

# SOFTWARE PROCESS ARCHITECTURE

## SYSTEM MODIFICATION SCENARIO - PHASES & SUBPHASES

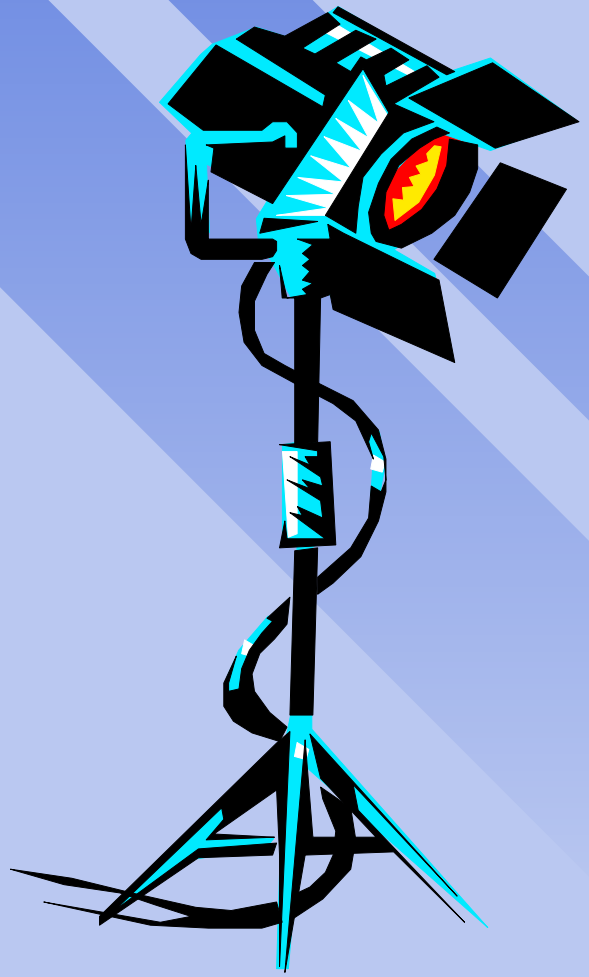


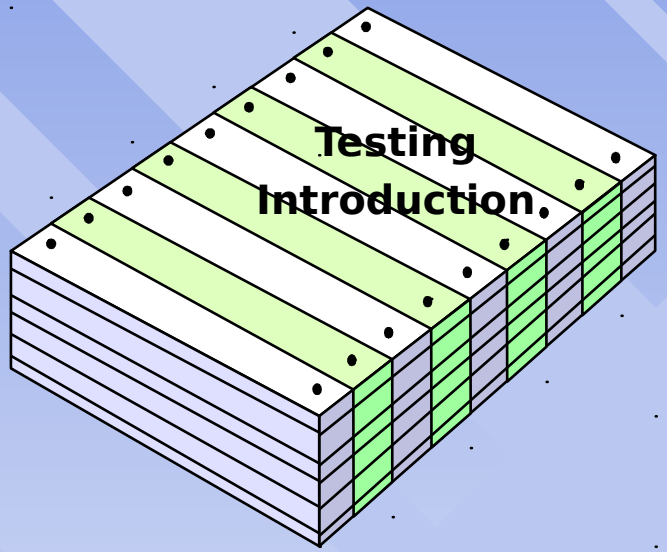
# Performance Check



**Open the Performance Check Package to SPI, CMM, SMS Review and answer the questions.**

# SECTION 2





# Testing Introduction

- The student will*:*
  - Identify mandatory and optional testing
  - Identify the various levels of testing
  - Identify testing responsibility within DFAS
  - Identify the various testing methods
  - Define Unit Test (UT)

# Testing Introduction

- What is Testing ?
  - “Program testing can be used to show the presence of bugs but never their absence.” Dijkstra 1969
  - “Testing is the process of establishing confidence that a program or system does what it is supposed to.” Hetzel 1973
  - “Testing is the process of executing a program or system with the intent of finding errors.” Myers 1979

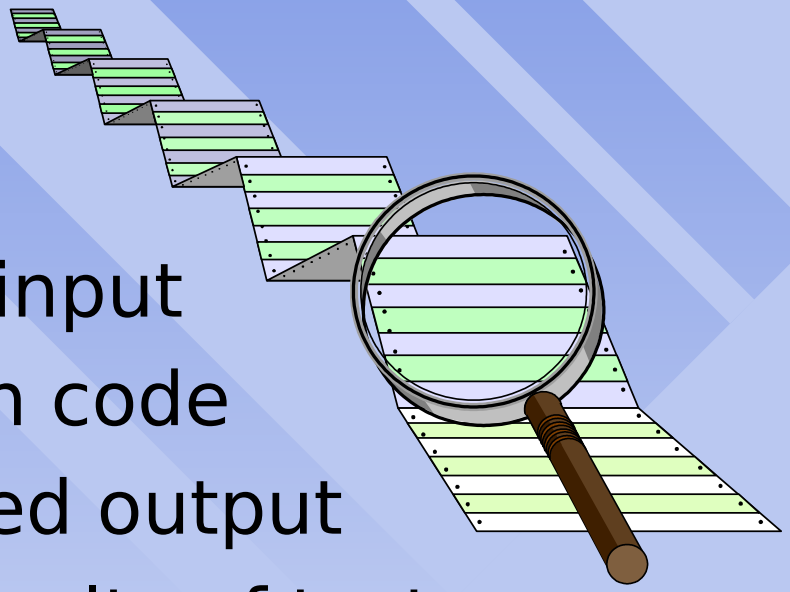
# Performance Check



**Open the Performance Check Package to SPI, CMM, SMS Review and answer the questions.**

# Testing Axioms

- Try to detect undiscovered defects
- Know when to stop
- Avoid on-the-fly tests
- Test invalid and valid input
- Impossible to test own code
- Must describe expected output
- Thoroughly inspect results of tests



# Testing Axioms

- As number of detected defects increases, the probability of the existence of more undetected defects also increases.
- Assign best programmers to test



# Proper Role of Testing

- Exhaustive testing is generally impossible
- Is the system a jungle or a cultivated garden?

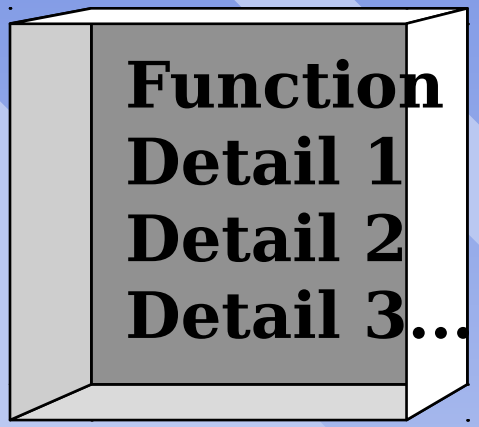


# Performance Check

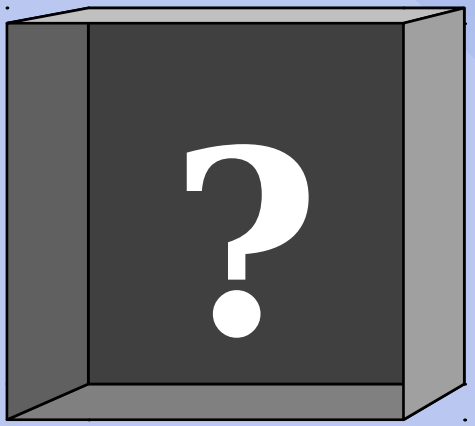


**Open the Performance Check Package to
Test Case Exercise
and answer the questions.**

# Basic Testing Strategies



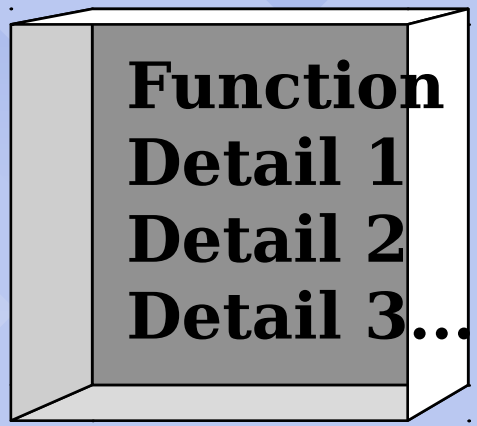

- Glass Box -
  - logic driven
  - knowledge of internal design
  - a.k.a., White Box




- Black Box -
  - data driven
  - functional requirements

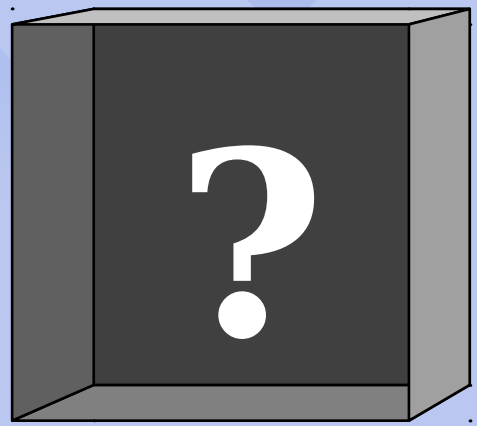
# Glass-Box Testing Methods

- Include
  - Statement Coverage
  - Decision (branch) Coverage
  - Condition Coverage
  - Path Coverage



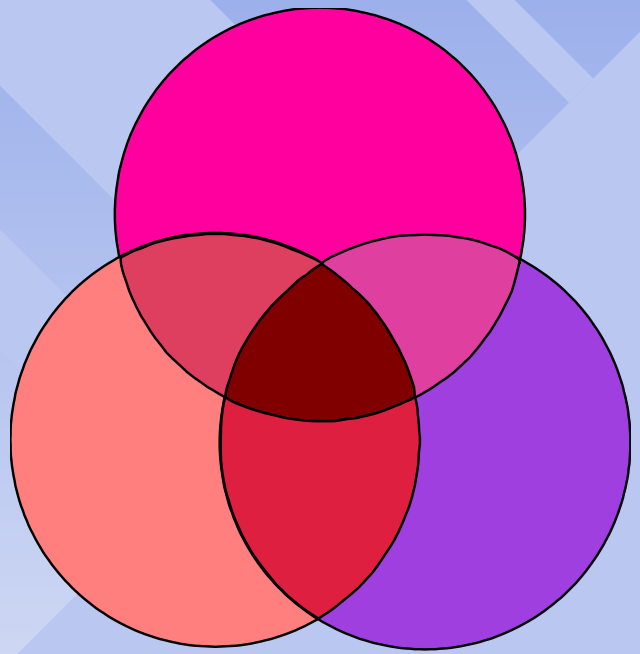
# Black-Box Testing Methods

- Commonly include:
  - Equivalence Partitioning
  - Boundary Value Analysis
  - Error Guessing
- Less-commonly include:
  - Cause-effect graphing
  - State Transition Testing



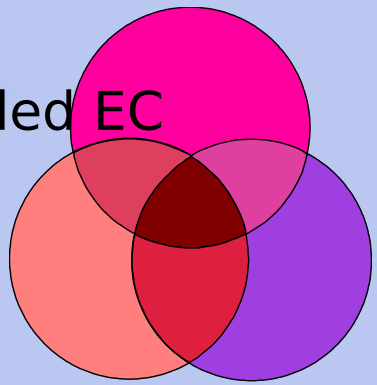
# Equivalence Partitioning

- For input conditions:
  - Proceduralizes identification of sets to test
  - Minimizes number of tests performed to cover the important sets



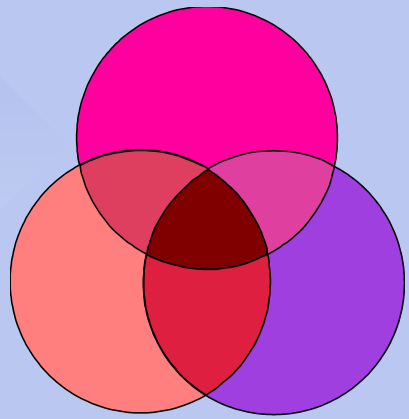
# Equivalence Partitioning

- First Step:
  - Identify Equivalence Classes (ECs)
    - » Ranges yield one valid and two invalid ECs
    - » Specific number of test values yield 1 valid and 2 invalid ECs
    - » Set of values yields 1 valid and 1 invalid
    - » Unique inputs yield 1 valid
    - » “Must be” yields 1 valid and 1 invalid
    - » Unique elements in an EC yield subdivided EC

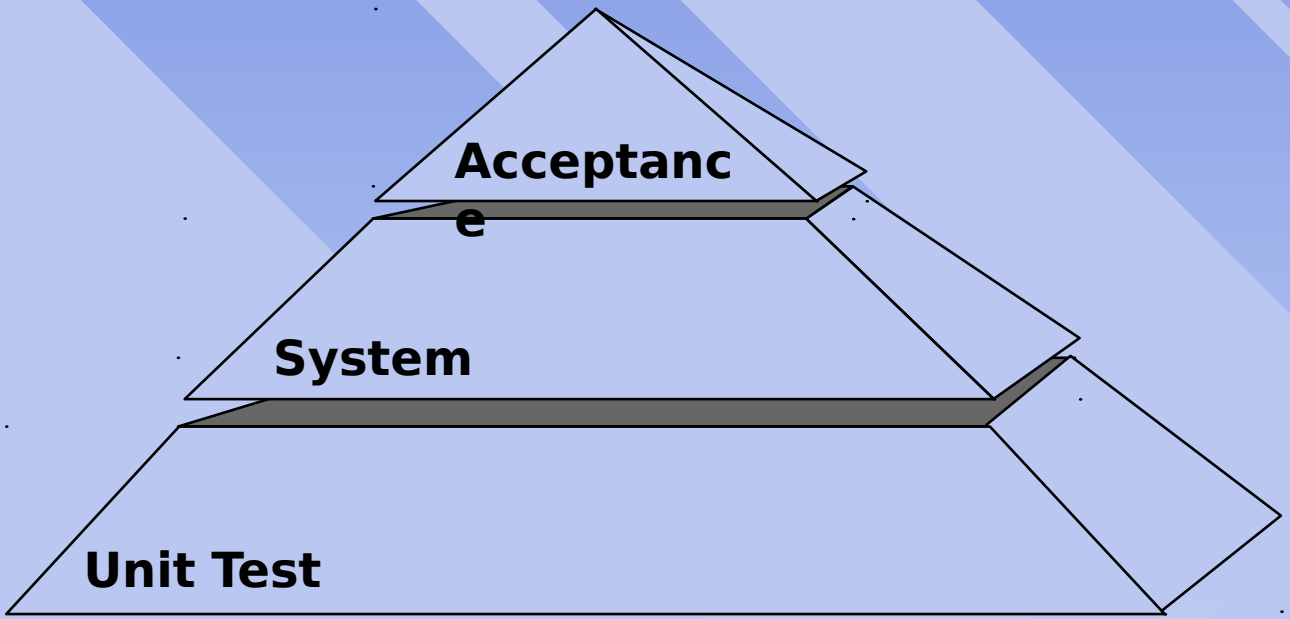


# Equivalence Partitioning

- Second Step:
  - Identify Test Cases:
    - » Assign unique number to each EC
    - » Until all valid ECs are covered, write new test cases covering as many valid ECs as possible.
    - » Until all invalid ECs are covered, write test cases that cover one and only invalid one EC.



# Testing Introduction



# Testing Introduction

## Testing Types

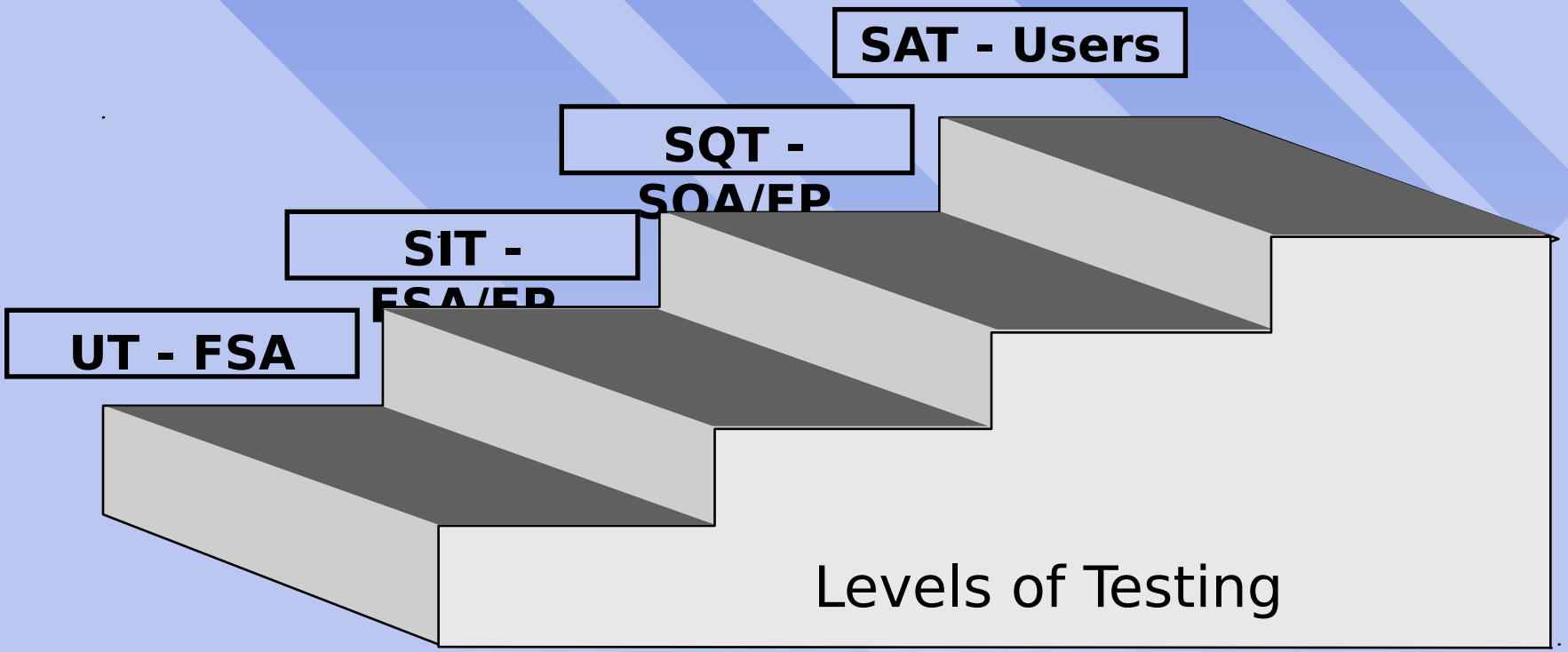
**Mandatory**

- Unit Test
- SIT
- SQT
- SAT
- Regression Test

**Optional**

- Prototype
- Desk Checking
- String Test
- Pilot Test
- Beta Test
- Parallel Test

# Testing Introduction



# Testing Introduction

- Testing Responsibility within DFAS

| | | | FSA | DMC | QA | FP | Users |
|---|---|---|---|---|---|---|---|
| Software Acceptance Test (SAT) | | | R | R | R | P/R | P |
| Software Qualification Test (SQT) | | | - | - | P | R | - |
| Software Integration Test (SIT) | P | | - | R | R | - | |
| Unit Test (UT) | P | - | R | R | - | | |

KEY:

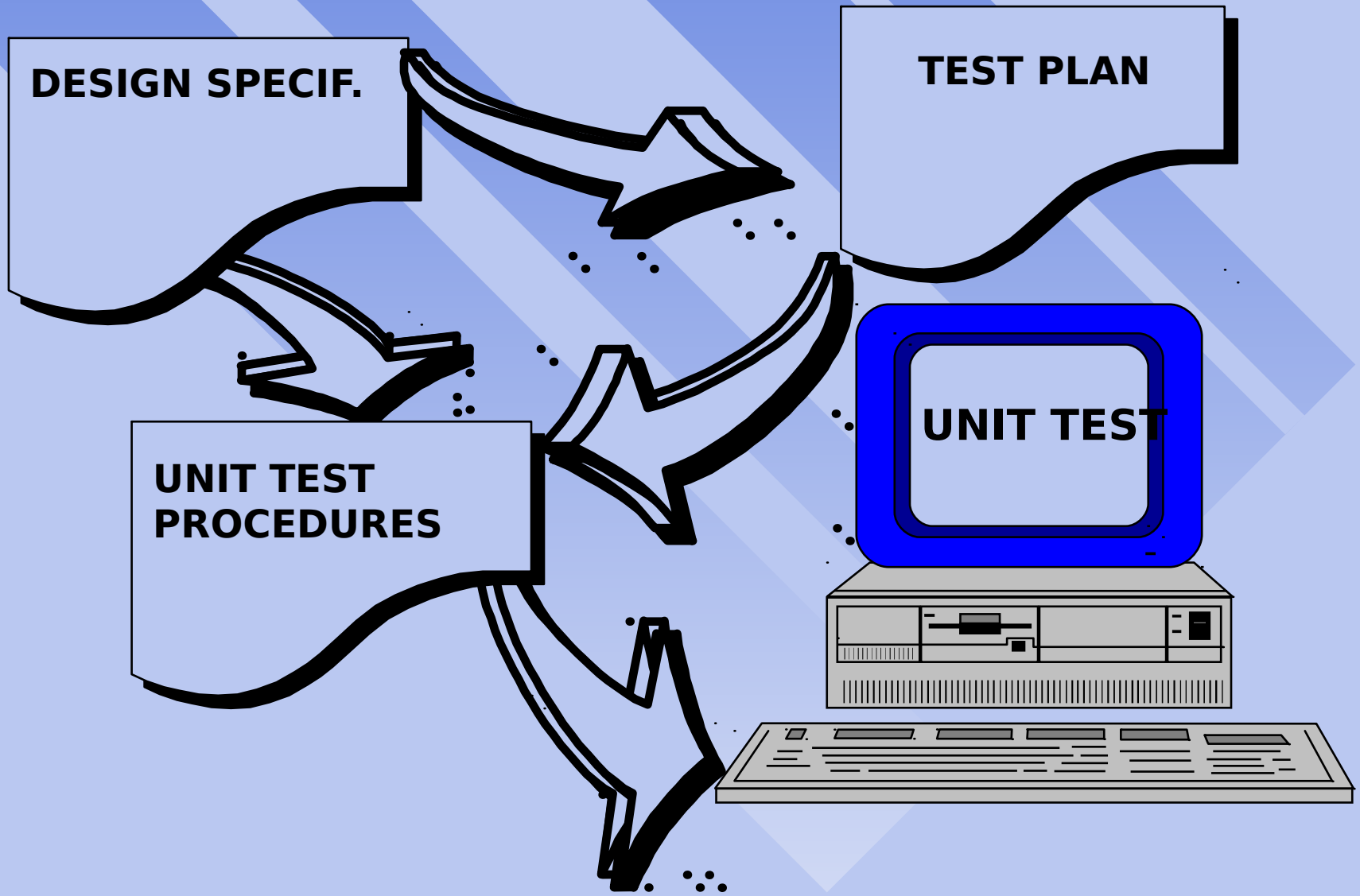
R = REVIEW Responsibility

P = PERFORM Responsibility

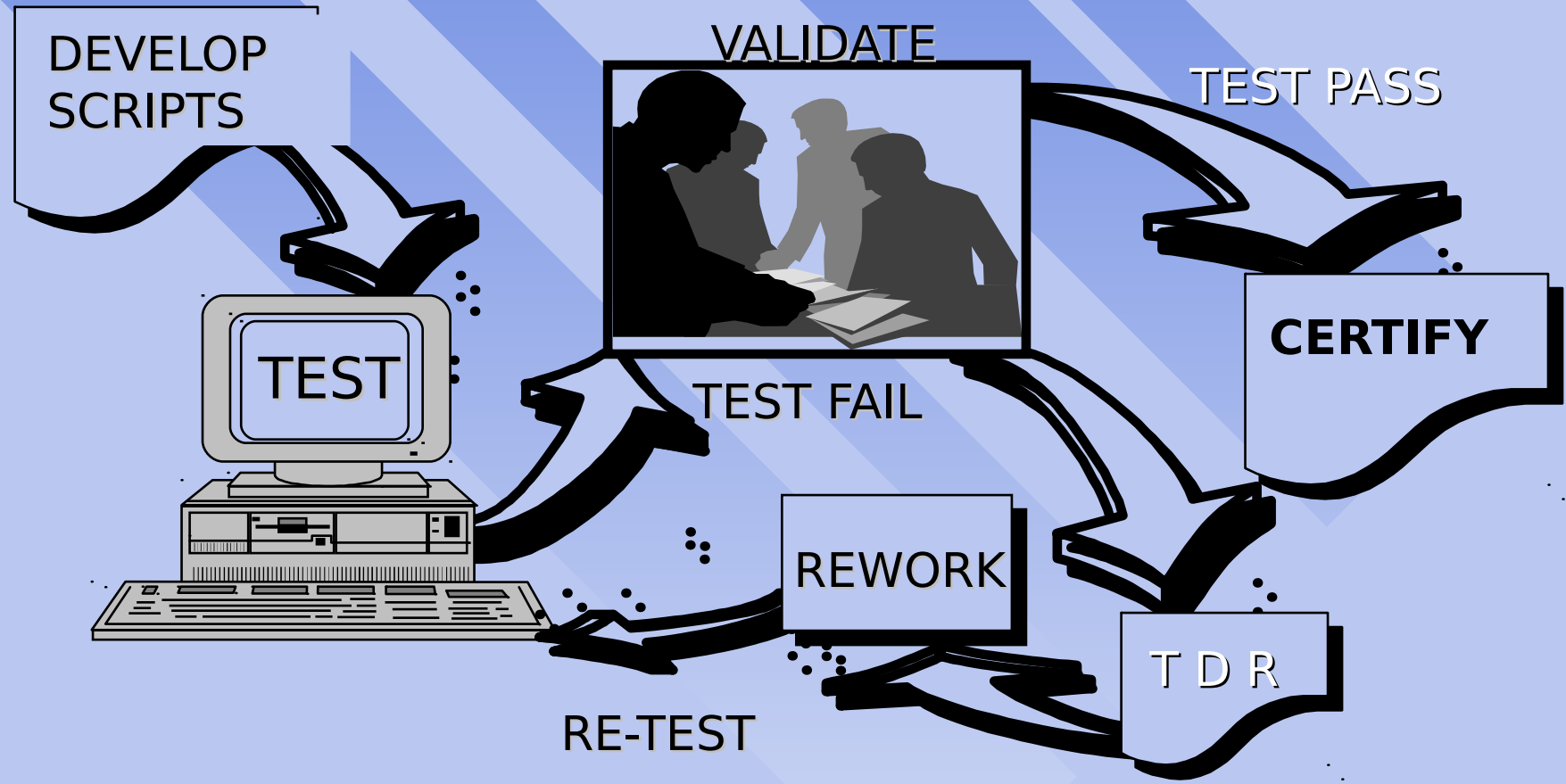
# Testing Introduction

- What is a Unit Test (UT) ? ? ?
  - Mandatory testing
  - Verification of software design
  - Lowest level software test
  - Programmer-executed and evaluated
  - Uses programmer-developed test data
  - Logic-driven (White Box) testing method

# UNIT TEST ACCEPTANCE



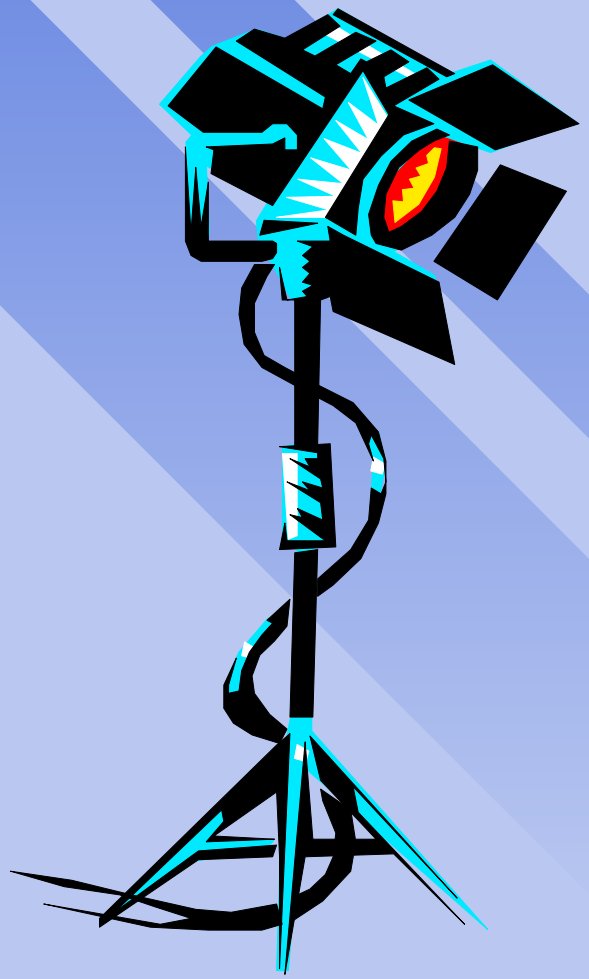
# TESTING SEQUENCE



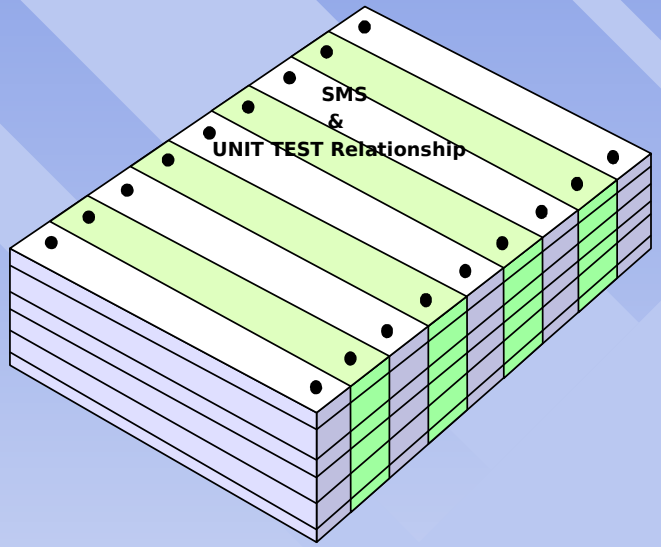
# Performance Check



**Open the Performance Check Package to Testing Introduction and answer the questions.**



# SECTION 3

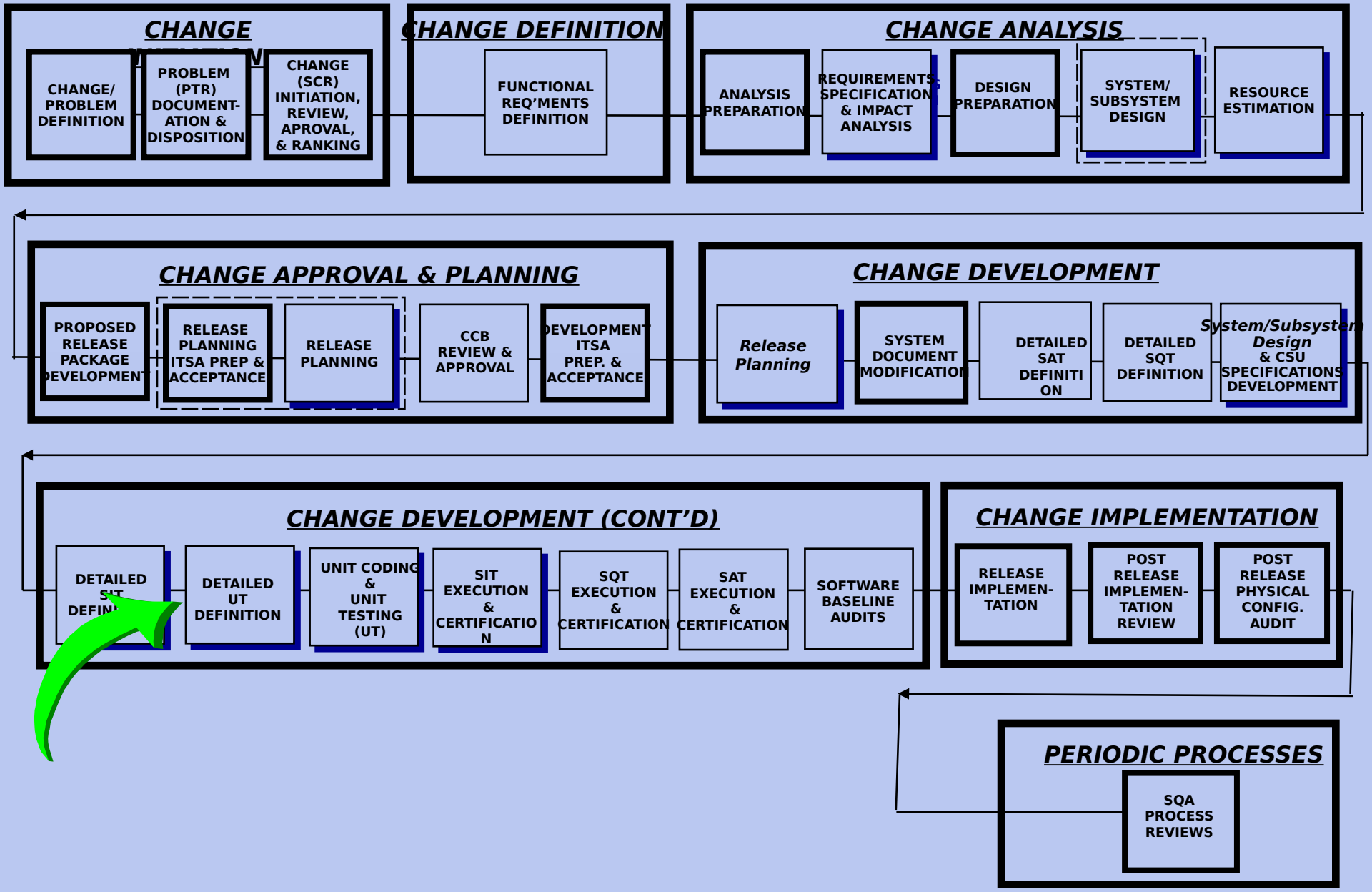


# SMS & UNIT TEST Relationship

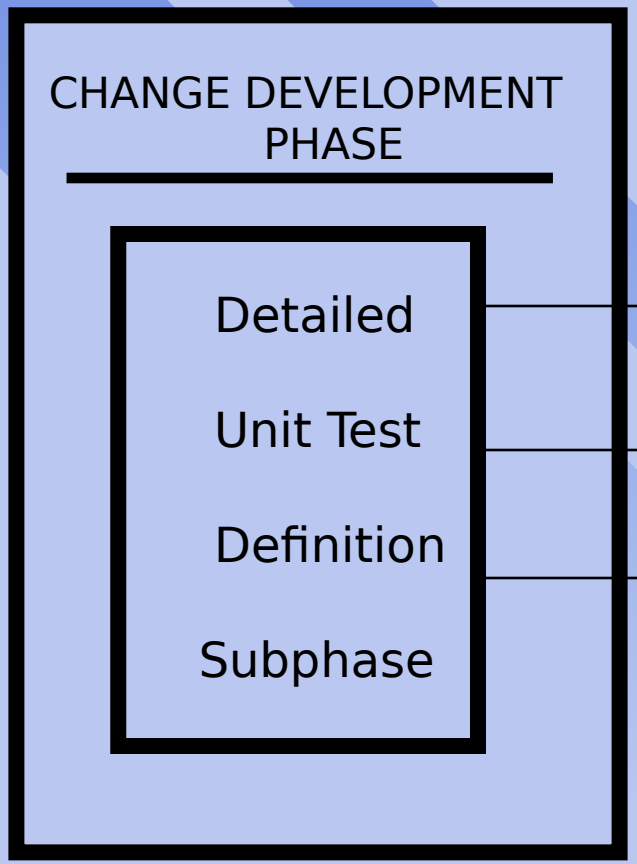
- OBJECTIVE:
  - Identify:
    - » Unit Test (UT) tasks
    - » Subtasks
    - » Procedures outlined in the SMS

# SOFTWARE PROCESS ARCHITECTURE

## SYSTEM MODIFICATION SCENARIO - PHASES & SUBPHASES



# 3 TASKS




Unit Test Infrastructure Modification

Unit Test Scripts Modification

Unit Test Data Modification

**Standard (s):**

**Mil Std 498 30 Mar 94**
**DFAS 5002.1-G**

**Input (s):**

**Test Infrastructure**

**Unit Test Infrastructure Modification Task (1 of 3)**

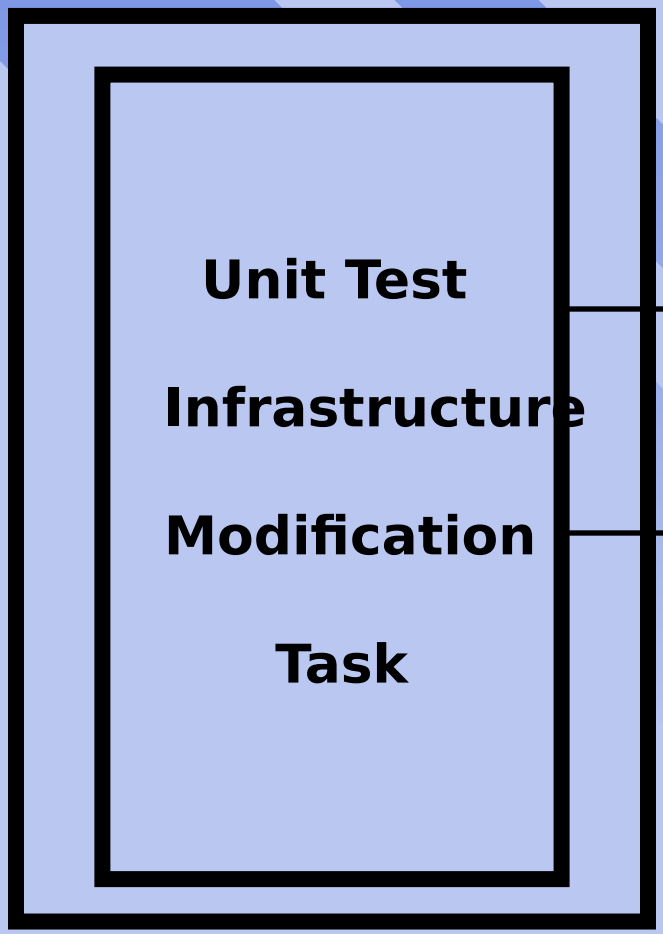
**Purpose: Define the structural blueprint and establish the test time dimensions.**

**Output (s):**

**Test Infrastructure**

**Skill(s):**

**Application Systems Programming**




## 2 Subtasks

- Unit Test Structural Blueprint Definition
- Unit Test Time Dimension Determination

# Infrastructure Modification Task

- Unit Test Structural Blueprint Definition Subtask (1 of 2)
  - Define the test master structure
  - Document the structural blueprint

# Infrastructure Modification Task

- Unit Test Time Dimension Determination Subtask (2 of 2)
  - Select date(s)/time(s) for the test
    - Cycles
    - Quarters
    - Year End
  - Document the time dimensions

# Performance Check



**Open the Performance Check Package to UT Infrastructure Modification and follow the directions.**

**Standard (s):**

**DFAS 5002.1-G**

**Input (s):**

**Test Scripts**

**Unit Test Scripts Modification Task (2 of 3)**

**Purpose: To develop specific unit test instructions for each test task**

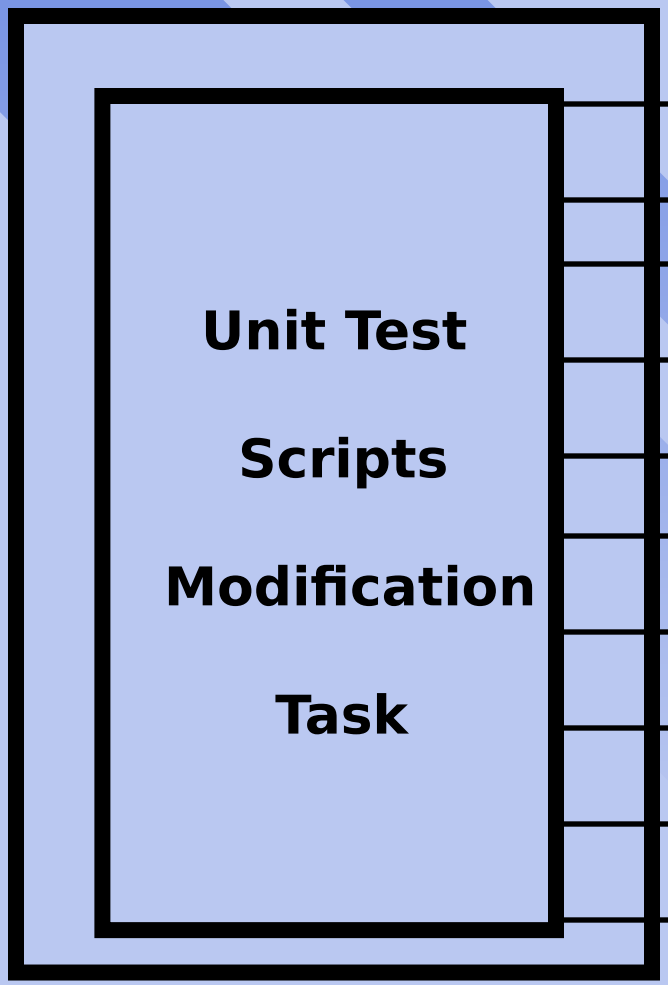
**Output (s):**

**Test Scripts**

**Skill(s)**

**Application Systems Programming**

# 10 Subtasks




- Test Case Modification Instructions
- Test Case Modification
- Screen Format Test Category Definitions
- Inquiry Test Category Definition
- Outputs & Updates Test Category Definitions
- Report Test Category Definition
- Input Test Category Definition
- Set-up Instructions Modification
- Execution Instructions Modification
- Evaluation Instructions Modification

# Test Scripts Modification Task

- Test Case Modification Instructions Subtask (1 of 10 )
  - Defines information to be modified for each category
    - » Functional
    - » Technical

# Test Scripts Modification Task

- Test Case Modification Subtask (2 of 10)
  - Develop narrative description
  - Define input transactions
  - Describe expected results
  - Define required file records

# Test Scripts Modification Task

- Screen Format Test Category Definition Subtask (3 of 10)
  - Identifies screen characteristics
  - Identifies cursor functions
  - Identifies menu bar functionality

# Test Scripts Modification Task

- Inquiry Test Category Definition Subtask (4 of 10)
  - Identify inquiry validation methods.
  - Define inquiry attributes to be tested.

# Test Scripts Modification Task

- Outputs and Updates Test Category Definition Subtask (5 of 10)
  - Defines criteria & specifications for verification and validation:
    - Calculations
    - Consistency
    - Certification

# Test Scripts Modification Task

- Report Test Category Definition Subtask (6 of 10)
  - Identifies report data standards
- Input Test Category Definition Subtask (7 of 10)
  - Defines transaction input criteria:
    - format
    - valid & invalid values
    - field & data checks

# Test Scripts Modification Task

- Set-up Instructions Modification Subtask (8 of 10)
  - Provides instructions to set the system to specific test point.
  - Executed only if needed

# Test Scripts Modification Task

- Execution Instruction Modification Subtask (9 of 10)
  - Define specific steps for executing the test
- Evaluation Instruction Modification Subtask (10 of 10)
  - Describe how to produce add'l docs.
  - Describe evaluation procedure
  - Describe wrap-up procedure
  - Update Requirements Traceability Matrix

# Performance Check



**Open the Performance Check Package to Unit Test Script Modification and follow the directions.**

## Standard (s):

**DFAS 5002.1-G**

## Input (s):

**Test Data**

**Unit test Scripts**

## Unit Test Data Modification Task (3 of 3)

**Purpose: To modify or develop the actual test data for the unit test.**

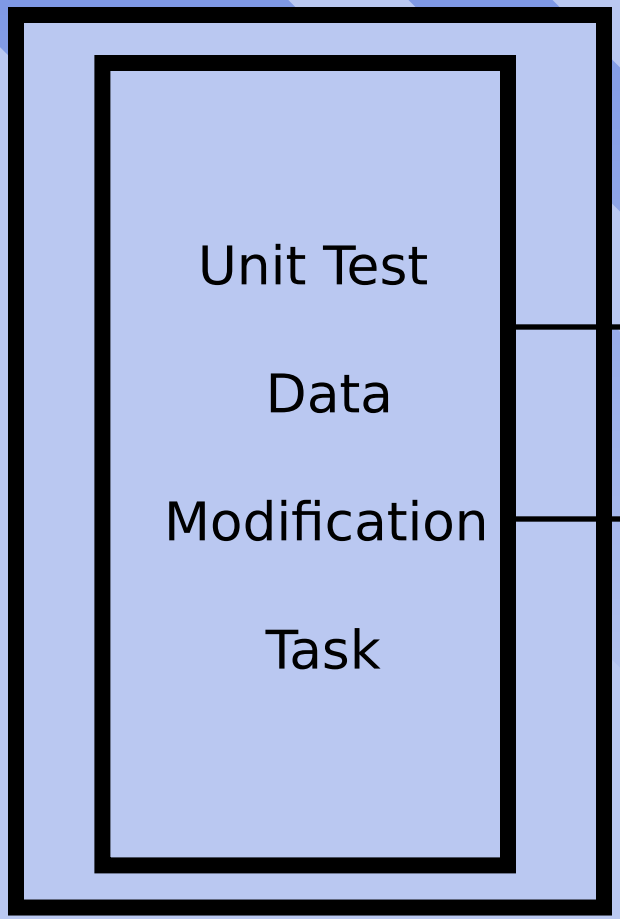
## Output (s):

**Modified Test Data**

## Skill(s):

**Application Systems Programming**

# 2 Subtasks




Test Tool Instructions Review **(OPTIONAL)**

Test Data Creation / Update Definition

# UT Data Modification Task

- Test Tool Instructions Review Subtask (1 of 2)
  - Review & follow test tool instructions
- Test Data Creation Update Definition Subtask (2 of 2)
  - Create actual test data
  - Update test data

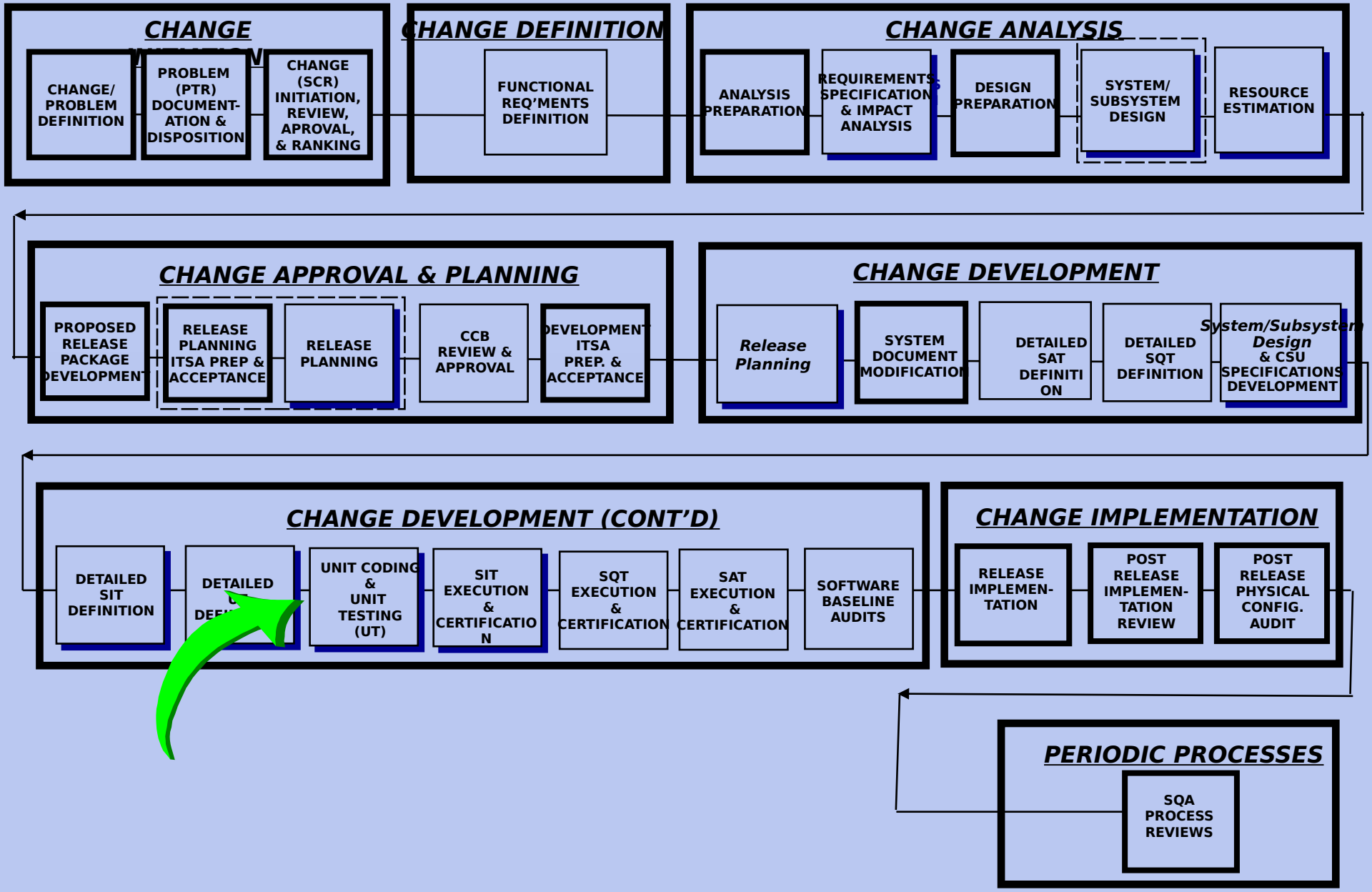
# Performance Check

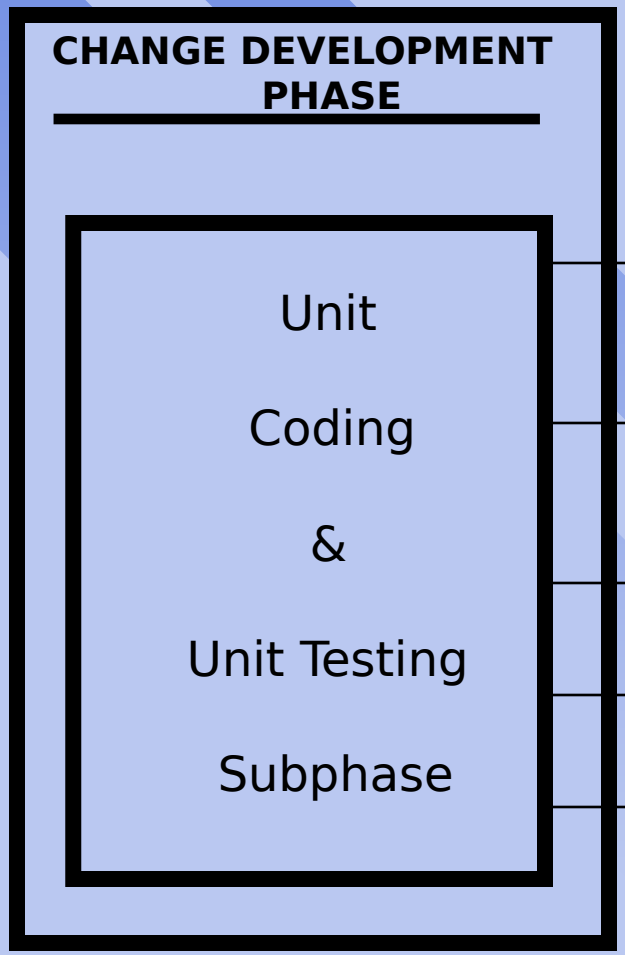


**Open the Performance Check Package to Unit Test Data Modification Task and answer the questions.**

# SOFTWARE PROCESS ARCHITECTURE

## SYSTEM MODIFICATION SCENARIO - PHASES & SUBPHASES






## 5 TASKS

- Unit Test Scripts, Data & Infrastructure Finalization
- Unit Test Testbed Data Initialization
- Unit Test Resource Modification
- Unit Test Execution
- Unit Test Evaluation

## Standard (s):

DFAS
5002.1-G

## Input (s):

Test Infrastructure

Test Data

Test Scripts

**Unit Test Scripts, Data & Infrastructure Finalization Task (1 of 5)**

**Purpose: To assure that every element of the planned unit test has been updated to reflect all current items and status.**

## Output (s):

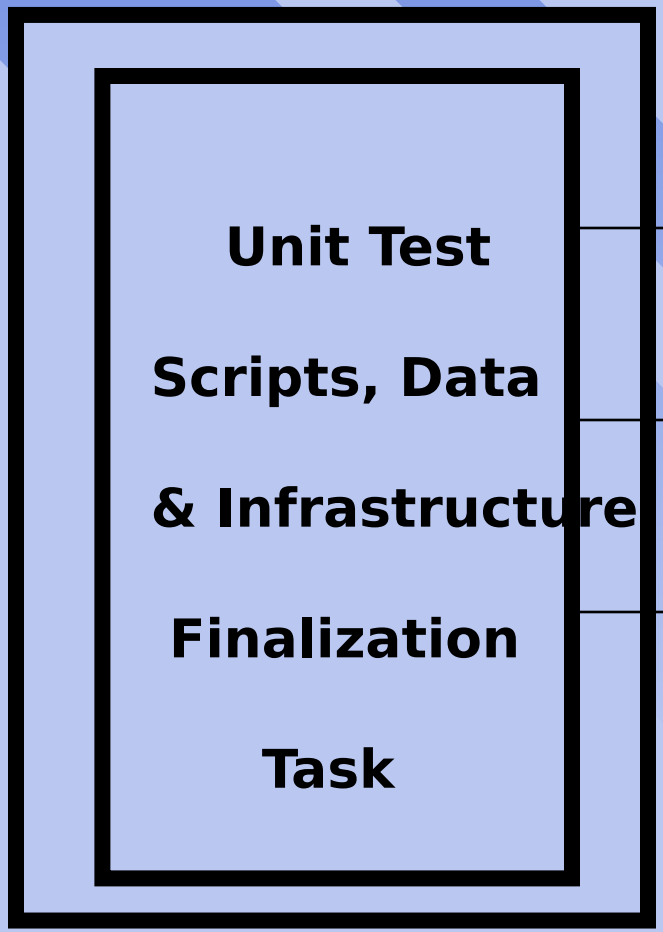
Finalized Test Infrastructure

Finalized Test Data

Finalized Test Scripts

## Skill(s) :

Application Systems Programming




## 3 Subtasks

- Test Infrastructure Finalization
- Test Scripts Finalization
- Test Data Finalization

# UT Finalization Task

- Test Infrastructure Finalization Subtask (1 of 3)
  - Review & update the test infrastructure
  - Structure blueprint/test time dimension processing sequence

# UT Finalization Task

- Test Scripts Finalization Subtask (2 of 3)
  - Review & update the test case.
    - narrative description
    - input transactions
    - expected results
    - required file records
  - Set-up, execution and evaluation instructions

# UT Finalization Task

- Test Data Finalization Subtask (3 of 3)
  - Review test data definitions
  - Update test data to reflect changes

# Performance Check



**Open the Performance Check Package to Unit Test Finalization Task and follow the directions.**

## Standard (s):

## Input (s):

Software Design Specification →

**Unit Test Testbed Data Initialization Task (2 of 5)**

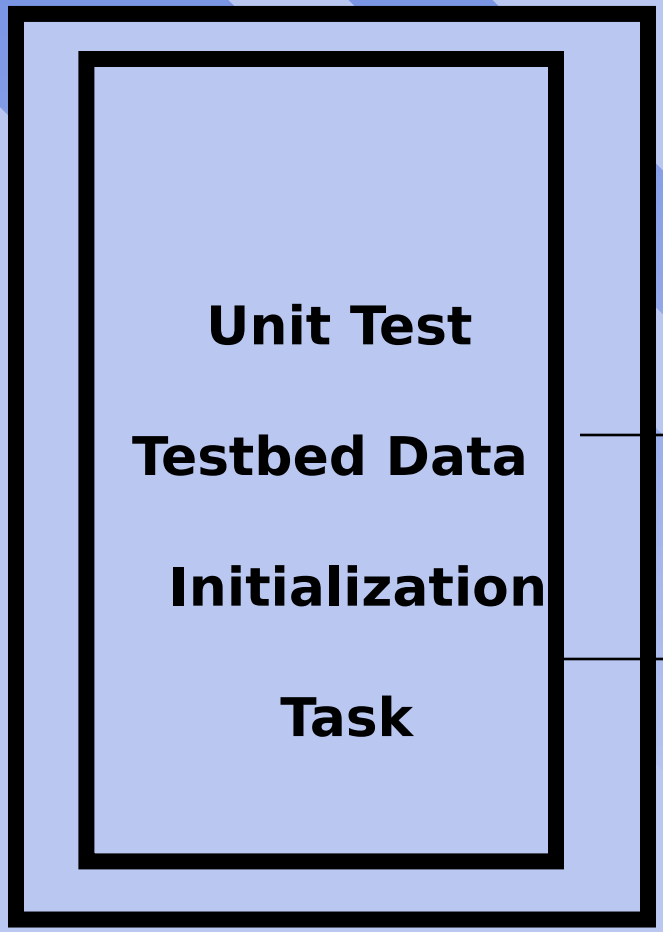
**Purpose: To ensure that all files are identified and loaded for the unit test**

## Output (s):

Populated Testbed →

## Skill(s) :

Application Systems Programming




## 2 Subtasks

- Test Save Point Determination
- Test Testbed Load

# Testbed Data Initialization Task

- Save Point Determination Subtask 1of 2
  - Review & determine appropriate save point for the unit test.
- Testbed Load Subtask 2 of 2
  - Load the files at the required point
    - » Database files
    - » Flat files

# Performance Check



**Open the Performance Check Package to Testbed Data Initialization Task and answer the questions.**

## Standard (s):

**DFAS 5002.1-G**

## Input (s):

**Support Resources**

**Unit Test Resource Modification Task (3 of 5)**

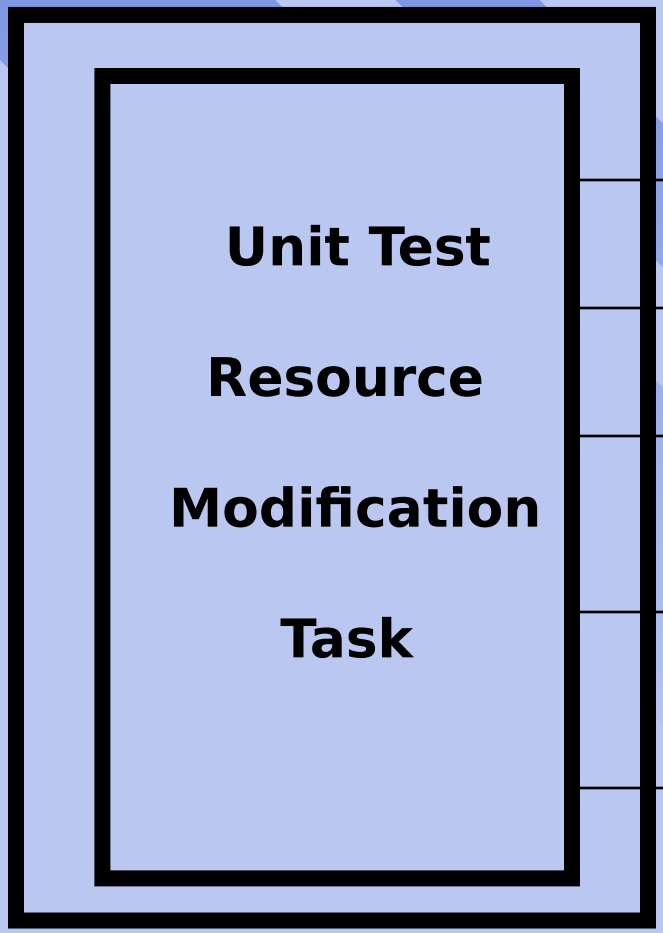
**Purpose: To ensure the availability of support resources.**

## Output (s):

**Modified Support Resources**

## Skill(s) :

**Application Systems Programming**




## 5 Subtasks

- Test JCL / ECL Preparation
- Test Utilities Acquisition
- Test Hardware Availability Confirmation
- Test Supporting Software Availability Confirmation
- Test Communications Availability Confirmation

# Resource Modification Task

- Prepare Test JCL/ECL Subtask (1of 5)
  - Prepare and maintain the test control language routines.
- Test Utilities Acquisition Subtask (2 of 5)
  - Identify Commercial Off-the Shelf (COTS) Software to be Purchased.
  - Identify Site-Unique Software (In-House)
  - Create “Drivers and Stubs”

# Resource Modification Task

- Availability Confirmation Subtasks (3 - 5)
  - Test Hardware
  - Test Supporting Software
  - Test Communications
    - » Identify hardware point of contact (POC).
    - » Contact and confirm the availability of hardware.

# Performance Check



**Open the Performance Check Package to UT Resource Modification Task and answer the questions.**

## Standard (s):

**DFAS 5002.1-G**

## Input (s):

- CSU
- CSCI
- Data Input
- JCL
- Test Infrastructure
- Test Data
- Test Scripts
- Testbed Data

**Unit Test Execution Task (4 of 5)**

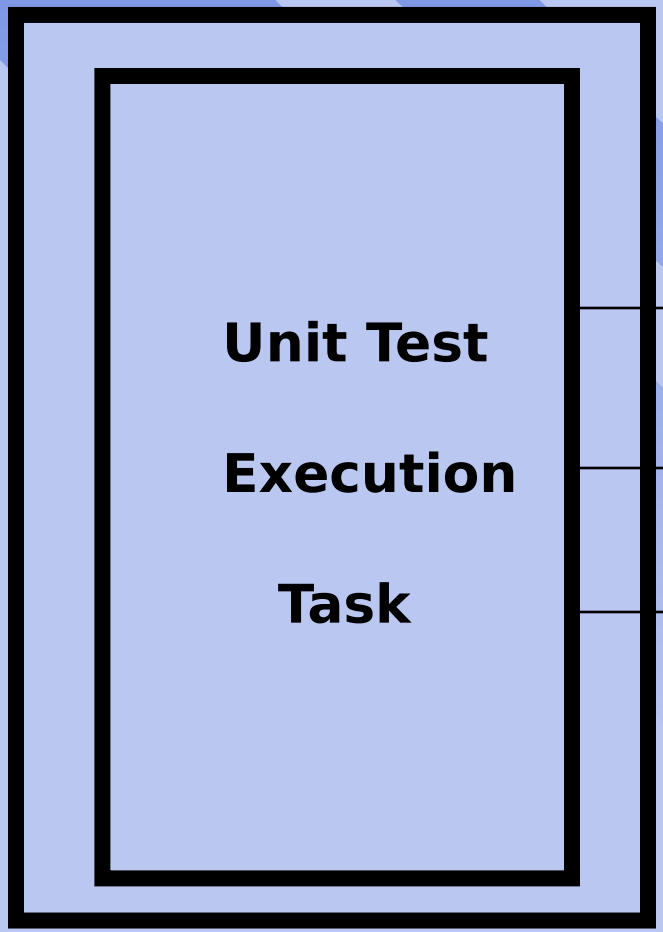
**Purpose: To execute the unit test.**

## Output (s):

- Execution Reports
- Testbed Data

## Skill(s) :

**Application Systems Programming**




## 3 Subtasks

- Test Set-Up Execution
- Test Execution
- Test Additional Output Production

# Unit Test Execution Task

- Test Set-Up Execution Subtask (1 of 3)
  - Move system from save point to test point
- Test Execution Subtask (2 of 3)
  - Execute the test as previously defined in the execution instructions

# Unit Test Execution Task

- Test Additional Output Production Subtask (3 of 3)
  - Additional output records are produced:
    - » Special reports
    - » Formatted database dumps
    - » Results of compare files

# Performance Check



**Open the Performance Check Package to Unit Test Execution Task and answer the questions.**

## Standard (s):

DFAS 5002.1-G
Mil Std 973, 17 April 92
ANSI / IEEE STD 1042-1987
DFAS Regulation 7920.3-R
CMIS Procedures Guide

## Input (s):

Test Results →

## Unit Test Evaluation Task (5 of 5)

**Purpose: To certify that the CSU has completed unit testing and is ready to release to SIT.**

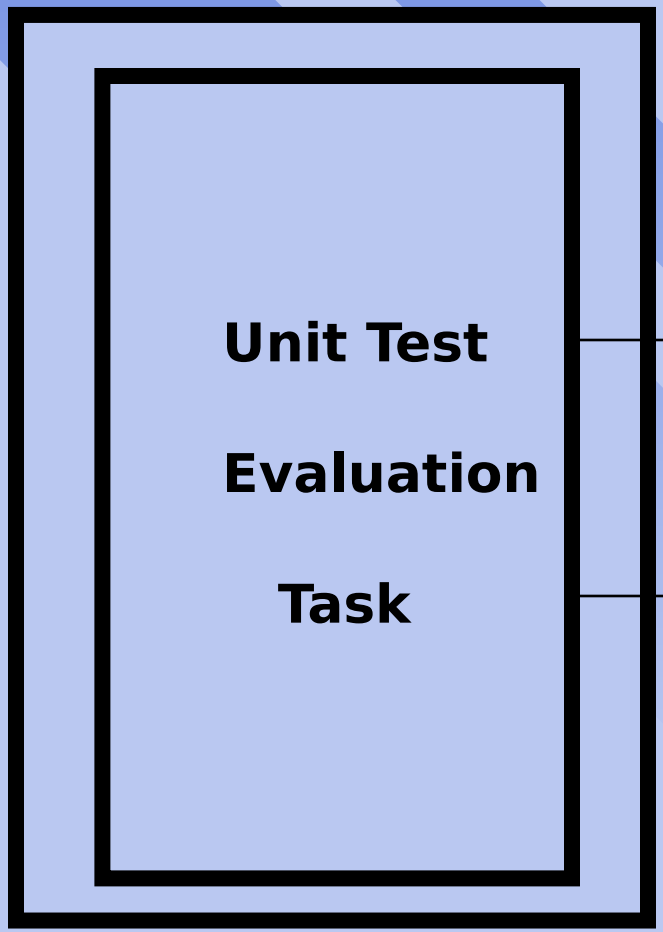
## Output (s):

CI Delivery Notice →

Test Results Certification →

## Skill(s) :

Application Systems Programming




## 2 Subtasks

- Test Results Analysis
- Test Certification

# Unit Test Evaluation Task

- Test Results Analysis Subtask (1 of 2)
  - Actual Test Results are compared to Expected Results
  - Programmer resolves inconsistencies.
- Test Certification Subtask (2 of 2)
  - Results are certified

# Performance Check



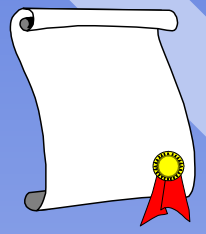
**OBJECTIVE:**

**Using the SMS and the Model System information, participant will be able to analyze and certify the test results.**

# Performance Check



**Open the Performance Check Package to
Unit Test Evaluation Task
and follow the directions and answer questions asked by the trainer.**



# CONGRATULATIONS

# YOU HAVE

# SUCCESSFULLY COMPLETED THE

# UNIT TEST COURSE